

# Roteamento de Dispositivos Móveis Utilizando a Tecnologia WIFI Direct

Ricardo Pagoto Marinho<sup>1</sup>, Urbano Botrel Menegato<sup>1</sup>, Ricardo Augusto Rabelo de Oliveira<sup>1</sup>

<sup>1</sup> Laboratório iMobilis - Departamento de Computação  
Universidade Federal de Ouro Preto – Ouro Preto, MG – Brazil

{ricardopmarinho, urbanobm, rrabelo}@gmail.com

**Abstract.** *Information exchange between mobile device grows every day. Although the communication rely on network with acces point (Wi-Fi, cellular, etc.). Using ad-hoc communication as Wi-Fi Direct, would avoid this dependence. Nowadays it doesn't supports multi-hop communication or moving devices. This work focuses on expand the use of Wi-Fi Direct tecnologia for the sent information of a device goes through others (multi-hop). We mesured the ammount of changed messages by devices using three routing algorithms: i) flooding, ii) AODV and iii) LAR . We could see that even with the challanges we can user routing algorithms over Wi-Fi Direct.*

**Resumo.** *A troca de informação entre dispositivos móveis cresce a cada dia. Porém essa comunicação depende de redes com ponto de acesso (Wi-Fi, celular, etc.). Utilizar comunicação ad-hoc, como Wi-Fi Direct, evitaria essa dependência. Atualmente ela não conta com suporte para comunicação multi-hop ou com os dispositivos se movendo. Este trabalho foca em expandir o uso da tecnologia Wi-Fi Direct para que a informação enviada por um dispositivo trafegue por vários outros (multi-hop). Medimos a quantidade de mensagens trocadas por vários dispositivos ao utilizarmos três algoritmos de roteamento: i) flooding, ii) AODV e iii) LAR. Pudemos perceber que, mesmo com os desafios encontrados, podemos fazer roteamento sobre Wi-Fi Direct*

## 1. Introdução

Os dispositivos móveis possuem um lugar de destaque na vida da sociedade atual. Várias pessoas possuem algum tipo destes dispositivos. Desta forma, a comunicação utilizando dispositivos móveis é muito frequente. Aplicativos que fazem com que eles se comuniquem são muito utilizados e se tornam populares. Porém estes aplicativos dependem de alguma rede para que as mensagens sejam trocadas, seja ela Wi-Fi ou celular.

Esses tipos de redes já são bem conhecidas e possuem garantias para que a informação seja entregue. Porém, algumas vezes não é possível utilizá-las. Redes Wi-Fi abertas de boa qualidade não são fáceis de achar ou nenhuma está disponível. Além isso, o dispositivo precisa estar no raio de alcance de um moden Wi-Fi para se conectar a ela. As redes celulares já são mais fáceis de serem acessadas, praticamente em qualquer lugar de uma cidade elas estão disponíveis. Porém o acesso a essas redes é feito mediante a um pagamento, seja ele prévio ou na hora do acesso.

As redes ad-hoc servem de alternativa mais baratas para a comunicação entre dispositivos. Neste tipo de rede os dispositivos se comunicam diretamente uns com os outros,

sem dependerem de pontos de acessos, como os modems nas redes Wi-Fi e as antenas nas redes celulares. Nesse contexto existe a tecnologia Wi-Fi Direct, desenvolvida pela Wi-Fi Alliance. Essa tecnologia utiliza a interface Wi-Fi do aparelho para que os dispositivos se comuniquem de forma ad-hoc.

Essa tecnologia é nova e poucos dispositivos possuem suporte a ela. Os dispositivos Android estão entre os que a suportam. A partir da versão 4.0 do sistema Android, dispositivos que o possuem são equipados com a tecnologia Wi-Fi Direct. Assim, podemos utilizar essa tecnologia para fazer a comunicação entre os dispositivos. Atualmente ela suporta apenas a comunicação entre dois dispositivos que estejam dentro do raio de alcance de ambos, sem saltos na comunicação, e os dispositivos devem estar parados para que a comunicação seja feita de forma eficiente.

[Botrel Menegato et al. 2014] utiliza a API de publicação de serviços para publicar informações. Informações como velocidade, localização e nível de bateria podem ser compartilhadas entre os dispositivos. Com base na implementação do Android e através de experimentos, observamos que existe um limite de 24 caracteres para o nome do serviço a ser publicado. Consideramos que 24 caracteres, em muitos casos, pode não ser suficiente para dar um nome adequado a um serviço. Além disso, nos testes de algoritmos de roteamento, em alguns casos, é necessário dividir a informação em mais de uma publicação.

Com este trabalho, pretendemos expandir a utilização da tecnologia em dispositivos móveis que utilizam Android fazendo com que eles sejam capazes de se comunicar em um ambiente em que a informação dê saltos (*multi-hop*) e a rede seja móvel (com entradas e saídas frequentes de dispositivos). Para isso utilizamos três algoritmos de roteamento para redes ad-hoc: i) *flooding*, ii) *Ad-hoc On-demand Distance Vector* (AODV) e iii) *Location-Aided Routing* (LAR). O algoritmo de *flooding* é o mais simples. Nele os dispositivos apenas enviam aos seus vizinhos, sem nenhum tipo de controle, mensagens para que eles estejam cientes de que o dispositivo está presente na rede. Os vizinhos por sua vez replicam sem nenhum tipo de controle essa informação, fazendo assim com que todos os dispositivos da rede fiquem sabendo dos outros. Nos algoritmos AODV e LAR os dispositivos que desejam se comunicar com algum outro na rede devem iniciar a etapa de descobrimento de rotas antes de enviar a mensagem. Nesta etapa, o dispositivo que deseja iniciar a comunicação envia uma requisição de rota a seus vizinhos. Esses, por sua vez, encaminham a mensagem até que ela chegue ao dispositivo destino, que responde à requisição, fazendo com que a resposta carregue a rota que a requisição percorreu até chegar ao destino, criando assim uma rota entre os dispositivos. O que difere o LAR do AODV é que o primeiro utiliza informações de localização e deslocamento do dispositivo destino para encaminhar as requisições.

Os testes feitos foram conduzidos de forma com que se monitorasse a quantidade de mensagens trocadas por cada algoritmo utilizando Wi-Fi Direct. Neles, os dispositivos estavam sempre ao alcance dos outros. Foram feitos testes com o número de dispositivos variando de 2 a 7 dispositivos. Os testes foram feitos para descobrir se a tecnologia é escalável e saber o comportamento da mesma a medida que novos dispositivos são inseridos na rede.

A contribuição deste trabalho é mostrar que, apesar das limitações da tecnologia,

é possível fazer roteamento ad-hoc de dispositivos Android utilizando Wi-Fi Direct.


## 2. Trabalhos Relacionados

[Barolli et al. 2010] propõe testes em MANET (*Mobile Ad-hoc Network*) utilizando o algoritmo de roteamento OLSR (Optimized Link State Routing). Utilizando 8 cenários diferentes para os testes eles coletaram informação sobre *throughput*, *Round-trip Time* (RTT) e perda de pacotes. Para tal seus testes foram de 150 segundos cada, em ambiente fechado com os dispositivos alcançando todos. Os testes foram feitos, utilizando o OLSR, sobre o fluxo de dados do TCP e UDP para contabilizar as métricas.

[Sharma et al. 2012] utilizam *Content Centric Networks* (CCN) para a comunicação entre dispositivos em uma MANET e aumentam a eficiência de entrega de mensagens nesta rede. As redes CCN são um paradigma que se preocupa apenas com o que a informação traz e não de onde ela vem. Eles, assim como [Barolli et al. 2010] também utilizam OLSR para conduzi-los testes. O algoritmo proposto se baseia em pontos de múltipla retransmissão. O trabalho utiliza probabilidades para que um nó seja selecionado dependendo se ele é um **pondo** de múltipla retransmissão ou não. Os testes foram feitos em dispositivos Android formando topologias de redes diferentes. Foram coletadas informações sobre a rede como taxa e tempo de entrega além de tráfego e *overhead*.

[Ikeda et al. 2011] propõem testes para avaliar o *throughput* e a taxa de perda de pacotes em ambiente de MANET utilizando os algoritmos OLSR e B.A.T.M.A.N.. Os dispositivos utilizados formaram dois cenários. Um no qual todos os dispositivos estavam parados e outro no qual um dos dispositivos se movia. Além disso, os dispositivos estavam em andares diferentes do prédio.

[Kim and Chung 2013] propuseram uma variação ao AODV. Eles utilizaram problemas em redes *mesh* sem fio (*Wireless Mesh Networks* – WMN) e com múltiplas interfaces e canais (multi-interface multi-channel -MIMC) e adaptaram o algoritmo para essas situações. O novo protocolo foi chamado de OM-AODV – *Optimized MIMC AODV*.

[Oki et al. 2013] fazem verificação da quantidade de bateria que os algoritmos AODV e OLSR consomem. Utilizaram 14 dispositivos para realizar os experimentos. O objetivo era verificar qual algoritmo era melhor utilizável em diferentes situações em dispositivos que utilizam energia solar como fonte. Os testes verificaram a eficiência desses dois algoritmos em situações com diferentes capacidades de transmissão e tamanho de informação. 

## 3. Algoritmos

Nesta seção, mostraremos como os dois algoritmos selecionados para os testes são propostos. Os algoritmos utilizados para os testes da tecnologia foram o algoritmo de *flooding*, o *Ad-hoc On-demand Distance Vector* – AODV e o *Location-Aided Routing* - LAR.

### 3.1. Flooding

No algoritmo de *flooding*, os nós participantes da rede apenas replicam as mensagens enviadas por seus vizinhos, sem nenhum processamento sobre elas. Desta maneira, dois nós que não sejam vizinhos, mas que possuem um vizinho em comum, podem se identificar como pertencentes à rede.

Porém, utilizando este algoritmo, temos garantia que a mensagem passará pelo melhor caminho entre dois nós, já que todos os nós da rede a receberão.

### 3.2. AODV

O algoritmo AODV é um algoritmo do tipo reativo, nos quais um nó só saberá se existe uma rota até um destino após uma fase de descobrimento de rotas.

A fase de descobrimento de rotas é iniciada assim que um nó  $s$  quer se comunicar com outro nó  $d$  na rede. Caso o nó  $s$  não possua rota para  $d$ , a fase de descobrimento de rota é iniciada. Primeiro o nó  $s$  envia a seus vizinhos, através de *broadcast*, uma mensagem RREQ (*Route Request*), que contém seu identificador, o identificador do nó  $d$  e a rota pela qual a mensagem passou (rota vazia quando da criação da mensagem). A mensagem é replicada pelos nós que a receberam através de *broadcast*. Estes nós adicionam seus identificadores à rota para que a resposta da requisição siga o caminho que ela percorreu. A resposta é criada em dois casos: i) a mensagem chega no nó  $d$  ou ii) a mensagem chega em um nó intermediário que possui uma rota para  $d$ . Nesses dois casos uma mensagem RREP (*Route Reply*) é criada. Essa mensagem possui o identificador de  $s$ , o destino final da rota,  $d$ , e a rota pela qual a mensagem RREQ passou para que chegasse até este nó e pela qual a mensagem RREP deve retornar até chegar ao nó  $s$ . No caso de um nó intermediário possuir uma rota para  $d$ , ele adiciona, além de seu identificador, a rota para  $d$  que ele possui no campo de rota da mensagem.

Caso algum dispositivo saia da rede, seja por problemas de conexão ou por ele realmente ter deixado a rede, uma mensagem RERR (*Route Error*) é criada. Ela é enviada pelo nó que identificou o erro para notificar aos integrantes da rede que o nó não está mais disponível. Assim que receber esta mensagem, o nó retira da tabela de roteamento toda rota que possui o nó problemático em uma rota  $d$ .

### 3.3. LAR

O algoritmo LAR, assim como o AODV, é um algoritmo reativo. Ele também se baseia nas mensagens de requisição (RREQ) e resposta (RREP) na formação de rotas. Porém, o que difere o LAR do AODV é a utilização de informações de posicionamento geográfico dos dispositivos na rede para fazer a comunicação.

Quando um dispositivo  $s$  deseja se comunicar com um dispositivo  $d$ , ele precisa primeiro das informações de onde o dispositivo  $d$  se encontrava em um instante de tempo  $t_2$ . Essas informações são posicionamento geográfico (latitude -  $X_d$  - e longitude -  $Y_d$ ), direção ( $D_d$ ) e velocidade ( $V_d$ ) de movimento. Com base nestas informações, quando  $s$  quiser formar uma rota para  $d$  em um instante  $t_2$ , ele calcula, de acordo com as informações de  $d$ , um possível espaço onde  $d$  possa estar em  $t_2$ . Esse espaço é um círculo centrado em  $X_d$  e  $Y_d$  e raio  $V_d(t_2 - t_1)$ .


Com esse espaço criado, as requisições são enviadas aos vizinhos de  $s$ . Quando um de seus vizinhos receber esta mensagem, ele verifica se ele está dentro do espaço especificado, caso esteja, continua propagando a requisição, caso não esteja, a descarta. A resposta ocorre da mesma maneira que no AODV.

Uma maneira de aumentar o espaço e fazer com que mais dispositivos na rede possam encaminhar a requisição é fazer com que o espaço seja um retângulo em que  $s$  e

a área em **qued** possa estar formada por uma diagonal. Dessa maneira o retângulo possuirá as pontas com as coordenadas  $(X_s, Y_s)$ ,  $(X_d + V_d(t_2 - t_1), Y_s)$ ,  $(X_d + V_d(t_2 - t_1), Y_d + V_d(t_2 - t_1))$  e  $(X_s, Y_d + V_d(t_2 - t_1))$ . Essas coordenadas englobarão o ponto onde  $s$  está ( $X_s$  e  $Y_s$ ) e o círculo onde possivelmente  $d$  está ( $X_d + V_d(t_2 - t_1)$  e  $Y_d + V_d(t_2 - t_1)$ ).

#### 4. Implementação

Para **conseguirmos** testar a tecnologia nos ambientes que propomos, utilizamos no Android, além das funções disponibilizadas para Wi-Fi Direct, as funções de publicação de serviços.

A tecnologia Wi-Fi Direct oferece funções para reconhecer dispositivos próximos que também a estão utilizando, como procura e identificação automáticas e eleger *Group Owner* (GO) para gerenciar a rede [Botrel Menegato et al. 2014 

Em seu trabalho, [Botrel Menegato et al. 2014] criou um *framework* de publicação de serviços para eleição de líderes de *cluster*. Desta forma eles pretendiam fazer com que a eleição do GO fosse feita de uma maneira mais confiável para que o eleito fosse relevante dentro do contexto do *cluster*.

Eles utilizaram todas as funcionalidades oferecidas pela API Android para Wi-Fi Direct e publicação de serviços, tais como, escanear a rede por outros dispositivos e receber mensagens enviadas por vizinhos. Deste modo, a informação sobre líderes de *cluster* era passada aos vizinhos dos nós para que eles pudessem decidir qual dispositivo seria o melhor indicado à tarefa.

As informações sobre os dispositivos na rede eram publicadas como serviços disponibilizados por eles. Essas publicações, na API, são strings com a informação que o dispositivo deseja enviar. Porém, ela possui limitações de tamanho, fazendo com que apenas pequenas quantidades de informações (por volta de 24 caracteres) possam ser enviadas de cada vez. Além disso, os serviços publicados pelos dispositivos são continuamente republicados pela API, parando somente quando explicitamente requisitado. Junta-se a isso, o fato de que existe um limite de serviços diferentes que podem ser publicados por um dispositivo. Cada um pode disponibilizar por volta de 7 serviços diferentes. Uma vez que esse número de serviços é publicado, é necessário parar a publicação de algum serviço para poder publicar outro.

[Botrel Menegato et al. 2014], porém, fazia **que** a informação fosse apenas transmitida para os vizinhos dos nós, sem que ela fosse retransmitida para outros nós pelos vizinhos. Isso abre novos usos do *framework*. Podemos utilizá-lo em cenários nos quais a informação deve ser passada por outros nós além dos vizinhos, como no caso de roteamento.

Como dito anteriormente, a API de publicação de serviços disponibilizada pelo Android possui limitações. Caso uma informação sobre o roteamento não caiba em apenas uma mensagem, ela é dividida em quantas necessárias para que a informação toda seja enviada. Juntando este fato ao de que os serviços continuam sendo republicados pela própria API, faz com que várias mensagens sejam criadas, trazendo uma carga maior à rede.

Desta maneira, a construção das mensagens (RREQ e RREP) tiveram que, além de conter as informações básicas dos algoritmos de roteamento, como origem, destino e rota,

dois números indicando quantas mensagens serão necessárias para enviar a informação e em qual estágio a atual mensagem está. Por exemplo, se uma mensagem dever ser dividida em duas, a primeira mensagem desta informação possuirá os números 2 (quantidade total de envios) e 1 (indicando que é a primeira mensagem). Já a segunda mensagem possuirá os números 2 e 2 (quantidade e segunda mensagem).

Quando um dispositivo receber uma mensagem com um número maior que 1 na quantidade total de mensagens a serem enviadas, ele guarda essa informação até que a segunda chegue. Assim que ela chegar, a informação é completada e armazenada.

Um problema que pode ocorrer é que essas mensagens podem chegar em ordem inversa. Caso isso aconteça a informação da segunda mensagem é armazenada e completada assim que a mensagem da primeira chegar.

Além disso, na implementação do LAR, que possui além das informações do AODV, informações de posicionamento e direção dos dispositivos, **tivemos** que dividir em mais mensagens para que todas as informações fossem enviadas, trazendo mais desafios para o controle das mensagens recebidas.

Para utilizar o trabalho de [Botrel Menegato et al. 2014], tivemos que expandir suas funcionalidades. Então foi necessário fazer que, quando um dispositivo receber uma requisição e ele tiver que reenviá-la, como discutido na Seção 3, ele publica essa informação como sendo um serviço que ele está disponibilizando, além de suas próprias requisições e respostas.

## 5. Experimentos

Os testes foram feitos para verificar a escalabilidade da tecnologia Wi-Fi Direct quando introduzimos mais dispositivos para se comunicar uns com os outros. Desta forma, foram feitos testes utilizando de 2 a 7 tablets, 5 Samsung Galaxy Tab 2, sendo que 3 possuem o sistema operacional Android versão 4.1.1 e 2 a versão 4.1.2, um Samsung Galaxy Tab 3 com sistema operacional Android 4.2.2 e um Samsung Galaxy Note com Android 4.3.3.

Os testes foram feitos com todos os tablets ao alcance de todos para que a escalabilidade pudesse ser verificada. Nos testes foram contabilizadas a quantidade de mensagens trocadas em cada um dos algoritmos citados anteriormente para verificar como a tecnologia se comporta com eles.

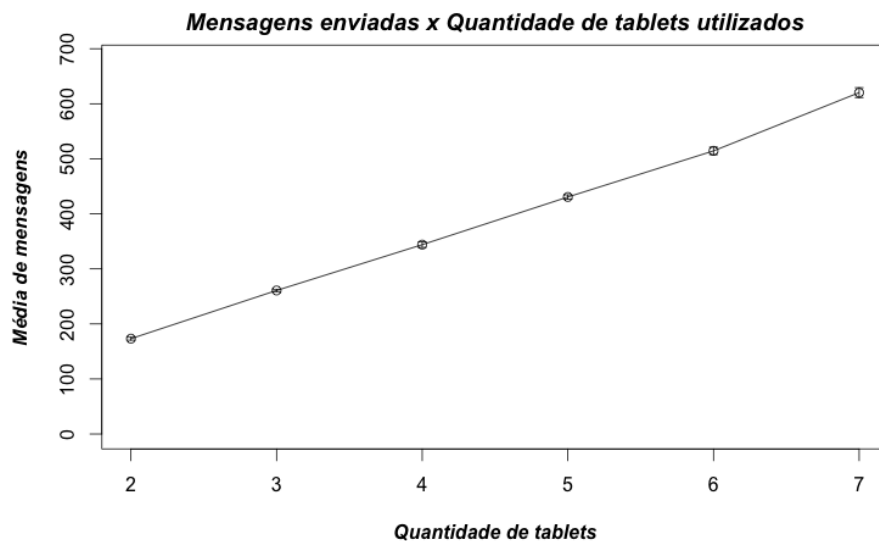
## 6. Resultados

Aqui os resultados das medições serão apresentados em forma de gráficos e serão discutidos.

### 6.1. Flooding

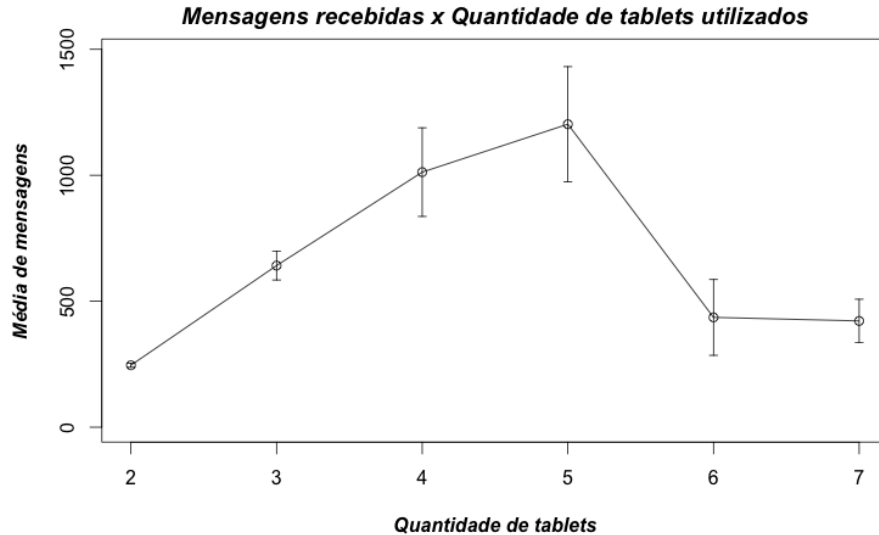
Na implementação do algoritmo de *flooding*, foi utilizada apenas um tipo de mensagem, a mensagem que o nó envia aos seus vizinho para notificá-los de sua existência na rede. Como discutido anteriormente, os vizinhos, ao receber esta mensagem a replicam aos seus vizinhos para que todos possam estar cientes dos nós participantes da rede. Desta maneira, a medição foi feita levando em consideração a quantidade de mensagens enviadas e recebidas por um determinado nó. A Figura 1 mostra a média de mensagens enviadas pelos tablets nos testes.





**Figura 1. Mensagens enviadas x Quantidade de tablets utilizados no *flooding***

Analisando aFigura 1, temos que a medida que a quantidade de tablets vai aumentando, a média de mensagens enviadas também aumenta, mostrando que a tecnologia, para enviar as mensagens, suporta a introdução de mais dispositivos na rede.

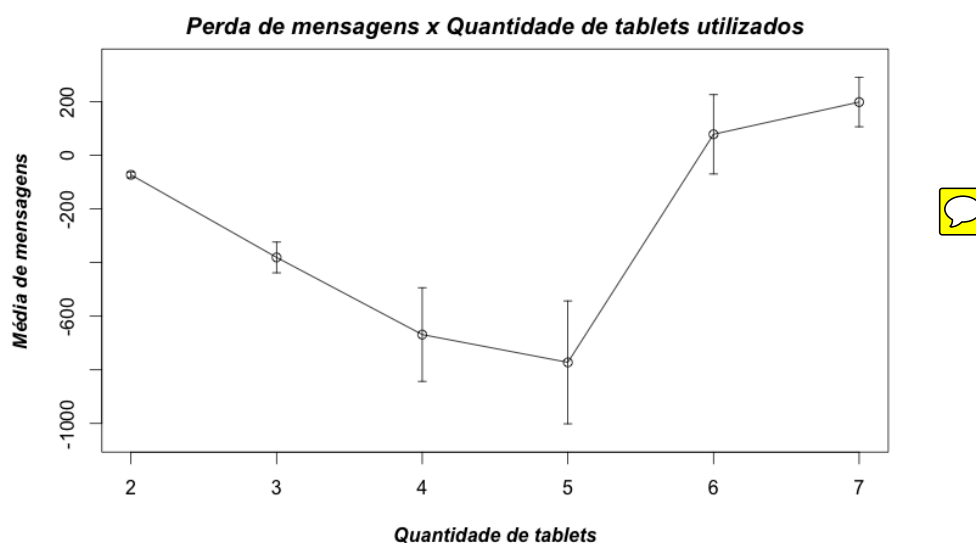


**Figura 2. Mensagens recebidas x Quantidade de tablets utilizados no *flooding***

Outro ponto que pode-se analisar é que esse aumento na média de envio é praticamente linear, mostrando que, até o ponto de medição, a tecnologia não altera seu comportamento diante da introdução de novos dispositivos. As figuras mostram também o desvio padrão ocorrido nas medições. Os desvios padrões são pequenos mostrando que a variação nas medições foram baixas, o que corrobora a afirmação anterior de que, para este tipo de medição, a tecnologia não altera seu comportamento diante da introdução de novos dispositivos.

Analisando a Figura 2 pode-se perceber que para uma rede de até 4 tablets, a média aumenta linearmente, mostrando que a tecnologia suporta sem problemas receber mensagens de 4 tablets simultaneamente. A partir daí a rede começa a se deteriorar, ou seja, perder a capacidade de receber mensagens, sendo que quando existem 6 e 7 tablets, a deterioração foi acentuada.

A deterioração da rede pode ser melhor mostrada na Figura 3. Como discutido anteriormente, a tecnologia envia várias mensagens até que o serviço seja retirado. Isso acarreta em um maior número de mensagens recebidas que enviadas.



**Figura 3. Perda de pacotes no *flooding***

Dito isso para a construção da Figura 3 foi feita a diferença entre mensagens enviadas e recebidas. Pelo o que foi discutido no parágrafo anterior esperasse que para até 5 tablets, mais mensagens sejam recebidas do que enviadas, logo a figura deverá ter um número negativo. Além disso, até este ponto, a diferença entre mensagens enviadas e recebidas deve aumentar, fazendo com que os números diminuam ainda mais (sejam mais negativos). Já para 6 ou mais tablets, ponto onde a rede degrada, este número deve ser positivo.

Como esperado, a diferença entre mensagens enviadas e recebidas foi crescendo até 5 tablets, a partir daí, pelo fato de a rede deteriorar, a diferença se inverte e mais mensagens são enviadas do que recebidas.

## 6.2. AODV

O algoritmo AODV possui 3 tipos de mensagens: i) RREQ, ii) RREP e iii) RERR. As mensagens RRRER não foram contabilizadas pois todos os tablets estavam no alcance dos demais, o que gera poucas dessas mensagens já que desconexões não são frequentes. As mensagens RREQ e RREP foram divididas em algumas categorias devido ao fato de nem todas as mensagens desse tipo serem relevantes para o dispositivo. Um exemplo disso é uma mensagem RREQ chegar em um dispositivo mas a requisição não é para ele. Assim as medições das mensagens RREQ foram divididas em mensagens enviadas



(RREQE), recebidas (RREQR) e mensagens recebidas nas quais o destino da mensagem era o dispositivo. Esta mensagem foi marcada como RREQ recebida do tipo 1 (RREQR1).

Nas mensagens RREP também ocorre o fato de nem toda mensagem ser relevante para o dispositivo que a recebeu. Então elas foram divididas em medições para mensagens enviadas (RREPE), recebidas (RREPR), mensagens na qual o dispositivo fez a requisição, que foram marcadas como sendo do tipo 1 (RREPR1) e mensagens na qual o dispositivo não fez a requisição mas faz parte da rota de encaminhamento da mensagem, marcadas como tipo 2 (RREPR2).

A Figura 4 mostra a comparação para os tipos de mensagens RREQ. A Figura ?? mostra que, com a introdução de mais tablets, a quantidade de mensagens RREQ enviadas aumenta. Isso mostra que, para este tipo de mensagem, a introdução de novos tablets não atrapalha o envio de mensagens.

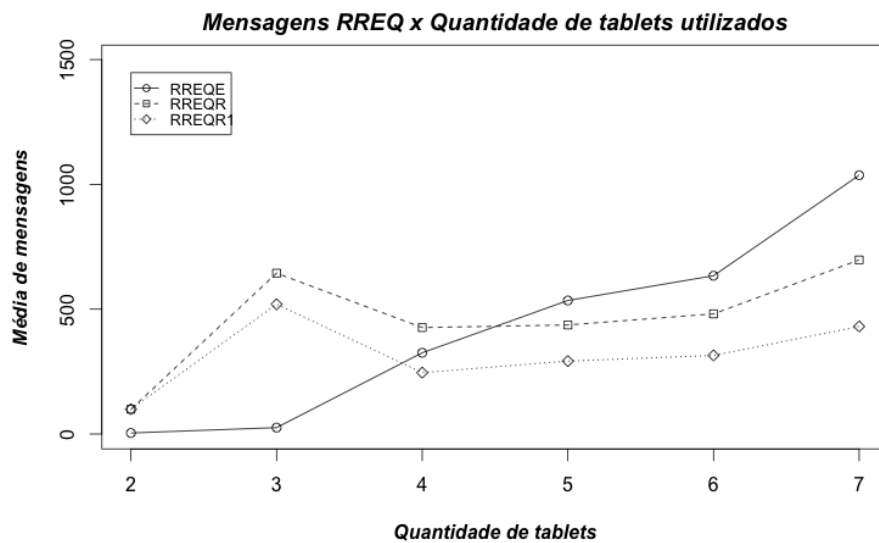
Para as mensagens RREQ recebidas vemos que a rede degrada a partir de 3 tablets. Neste ponto os tablets recebem a maioria das mensagens RREQ enviadas, porém a partir daí, a recepção destas mensagens cai. Um ponto interessante é que com 7 tablets o número de mensagens recebidas é quase igual ao número com 3 tablets, o que pode mostrar um limite da tecnologia. Para realmente confirmar esta afirmação, testes com mais tablets são necessários.

As mensagens RREQ recebidas do tipo 1 possuem o mesmo comportamento das mensagens RREQ recebidas. Isso ocorre pois as mensagens RREQ recebidas do tipo 1 estão contidas nas mensagens RREQ recebidas.

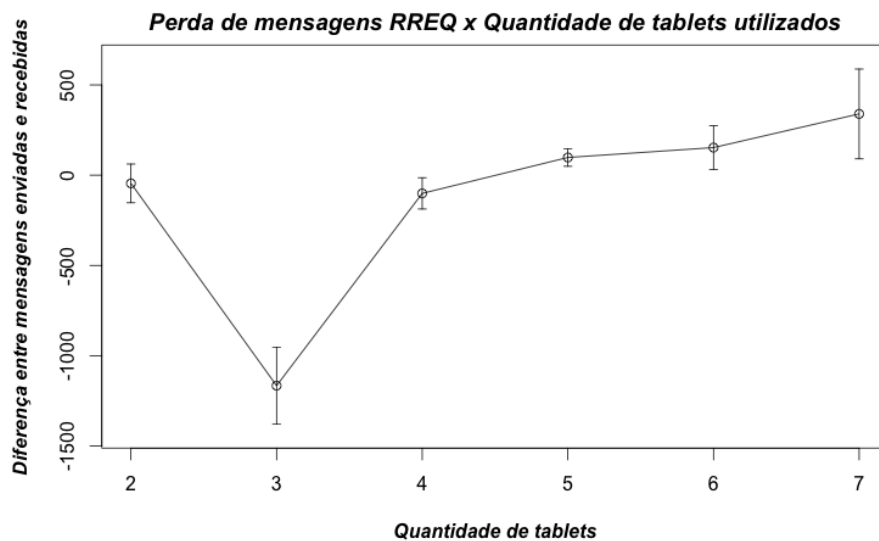
Comparando as curvas na Figura 4 vemos que ao introduzirmos mais do que 4 tablets, a rede deteriora. Enquanto a curva de enviadas sempre cresce, a de recebidas cai para 4 tablets e vai crescendo devagar para mais tablets.

A Figura 5 mostra a diferença entre mensagens RREQ enviadas e recebidas. É esperado que para 2 e 3 tablets, a curva cresça negativamente e para 4 ou mais tablets ela comece a crescer positivamente por causa da deterioração da rede.

Como discutido anteriormente, para 2 e 3 tablets a tecnologia recebe a maioria das mensagens e de suas cópias gerada pela API de serviços. A partir de então, essa relação começa a se inverter, até que com 5 tablets, a quantidade de mensagens enviadas é maior que a de recebidas, mostrando que a rede deteriorou.



**Figura 4. Comparação entre a quantidade de mensagens RREQ trocadas no AODV**

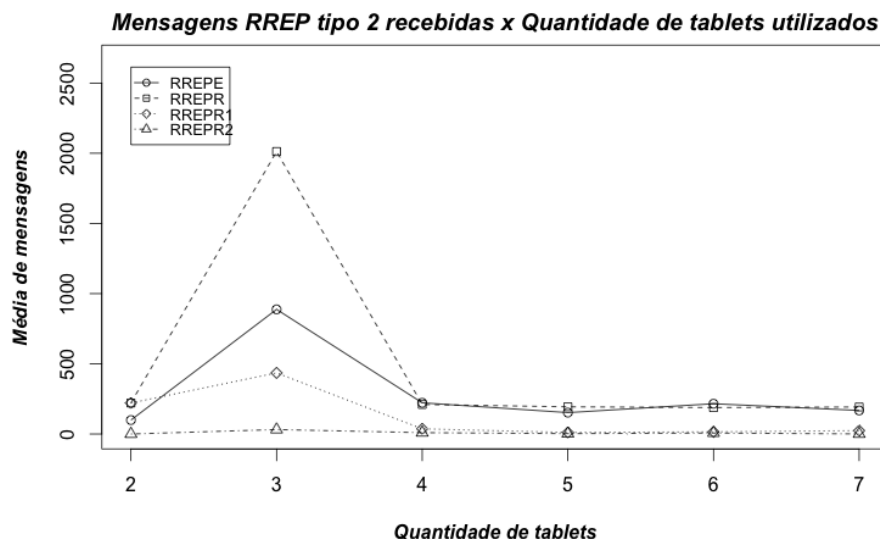


**Figura 5. Perda de mensagens RREQ x quantidade de tablets no AODV**

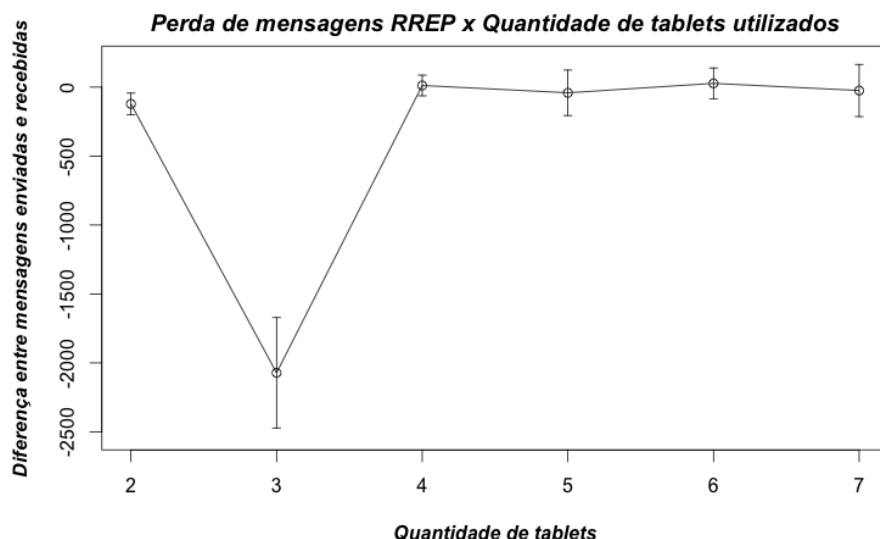
As Figuras 6 e 7 mostram as mesmas comparações feitas com as mensagens RREQ para as mensagens RREP.

Analizando a Figura 6 vemos que para todos os tipos de mensagens, a média delas diminui em 4 tablets. O que difere, principalmente falando de mensagens enviadas, das medições do *flooding* e das RREQ. Isso se dá pelo fato de as mensagens RREP serem produzidas somente após o recebimento de uma mensagem RREQ e como isso é afetado a partir de 3 tablets, afeta o envio das mensagens RREP. Além disso, a Figura 6 mostra que com 4 ou mais tablets, a quantidade de mensagens enviadas é praticamente igual à de recebidas. Isso nos diz que com 4 tablets a rede começa a deteriorar e o recebimento de mensagens é prejudicado. A Figura 7 mostra essa comparação de uma maneira melhor, fazendo a diferença entre a quantidade de mensagens enviadas com a de recebidas.

A Figura 7 nos mostra que a rede recebe a maioria das mensagens com até 3 tablets, com isso, o valor é o menor da medição. A partir deste ponto, a rede começa a se degenerar e a diferença nas comparações começam a ficar bem próximas. O comportamento desta figura difere do comportamento das outras figuras de perda pois esse tipo de mensagem depende do recebimento das mensagens de RREQ.



**Figura 6. Comparação entre quantidade de mensagens RREP trocadas no AODV**



**Figura 7. Perda de mensagens RREP x quantidade de tablets no AODV**

### 6.3. LAR

Nos testes com o LAR, percebemos que o Wi-Fi Direct parava de receber e enviar informações sobre o roteamento muito rapidamente (cerca de 1 minuto). Isso nos levou a fazer considerações sobre o motivo.

Devido ao fato de a API possuir limitações, como quantidade de serviços que podem ser disponibilizados, fizemos testes limitando essa quantidade para determinarmos quanto tempo o Wi-Fi Direct se manteria funcionando com um certo número de serviços. Os resultados estão na Tabela 1

**Tabela 1. Quantidade de serviços x tempo de funcionamento no LAR**

Quantidade de Serviços	Tempo (min)
Sem limitação	1
10	5
8	15
6	15

Essa tabela nos mostra que um possível limite de tempo de funcionamento do Wi-Fi Direct quando usado para roteamento é por volta de 15 minutos. Este valor foi observado também nos testes com o AODV, enquanto que no *flooding* foi encontrado um valor de 20 minutos. Isso nos mostra que a quantidade de serviços disponibilizados ao mesmo tempo pela tecnologia, afeta diretamente seu tempo de uso.

## 7. Conclusão

Analisando os resultados e os gráficos, conclui-se que, para redes pequenas, com 2 ou 3 dispositivos, a tecnologia suporta a carga de se fazer roteamento. Porém, com mais dispositivos, o roteamento fica prejudicado por causa da carga trazida à rede. Contudo, mesmo com esses problemas, é possível fazer roteamento de dispositivos em uma rede ad-hoc utilizando a tecnologia Wi-Fi Direct.

Além disso, descobrimos que a tecnologia Wi-Fi Direct é afetada a medida que publicamos mais serviços diferentes ao mesmo tempo, fazendo com que ela pare de publicar e receber informações.

Uma contribuição tecnológica é sugerir um aumento do número de caracteres para a publicação de serviços.

## 8. Referências

### Referências

- Barolli, L., Ikeda, M., Xhafa, F., and Duresi, A. (2010). A testbed for manets: Implementation, experiences and learned lessons. *Systems Journal, IEEE*, 4(2):243–252.
- Botrel Menegato, U., Souza Cimino, L., Delabrida Silva, S. E., Medeiros Silva, F. A., Castro Lima, J., and Oliveira, R. A. R. (2014). Dynamic clustering in wifi direct technology. In *Proceedings of the 12th ACM International Symposium on Mobility Management and Wireless Access, MobiWac '14*, pages 25–29, Montreal, QC, Canada. ACM.
- Ikeda, M., Kulla, E., Hiyama, M., Barolli, L., and Takizawa, M. (2011). Experimental results of a manet testbed in indoor stairs environment. In *Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on*, pages 779–786.

- Kim, W.-S. and Chung, S.-H. (2013). Design of optimized **aodv** routing protocol for multi-interface multi-channel wireless mesh networks. In *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*, pages 325–332.
- Oki, O., Mudali, P., Mutanga, M., and Adigun, M. (2013). A testbed evaluation of energy-efficiency of routing protocols in battery-powered wireless mesh networks. In *AFRICON, 2013*, pages 1–7.
- Sharma, P., Souza, D., Fiore, E., Gottschalk, J., and Marquis, D. (2012). A case for manet-aware content centric networking of smartphones. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, pages 1–6.